Abstract interpretation and types for systems biology

François Fages, Sylvain Soliman

Abstract Interpretation and Types for Systems Biology

François Fages, Sylvain Soliman *

Project-team Contraintes, INRIA Paris-Rocquencourt, BP105, 78153 Le Chesnay Cedex, France.

Abstract

Abstract interpretation is a theory of abstraction that has been introduced for the analysis of programs. In particular, it has proved useful for organizing the multiple semantics of a given programming language in a hierarchy corresponding to different detail levels, and for defining type systems for programming languages and program analyzers in software engineering. In this paper, we investigate the application of these concepts to systems biology formalisms. More specifically, we consider the Systems Biology Markup Language SBML, and the Biochemical Abstract Machine BIOCHAM with its differential, stochastic, discrete and boolean semantics. We first show how all of these different semantics, except the differential one, can be formally related by simple Galois connections. Then we define three type systems: one for checking or inferring the functions of proteins in a reaction model, one for checking or inferring the activation and inhibition effects of proteins in a reaction model, and another one for checking or inferring the topology of compartments or locations. We show that the framework of abstract interpretation elegantly applies to the formalization of these further abstractions, and to the implementation of linear or quadratic time type checking as well as type inference algorithms. Furthermore, we show a theorem of independence of the graph of activation and inhibition effects from the kinetic expressions in the reaction model, under general conditions. Through some examples, we show that the analysis of biochemical models by type inference provides accurate and useful information. Interestingly, such a mathematical formalization of the abstractions commonly used in systems biology already provides some guidelines for the extensions of biochemical reaction rule languages.

Key words: SBML, type inference, hierarchy of semantics, ordinary differential equations, Markov chains

* Corresponding author. This article is an extended version of [19].
Email addresses: Francois.Fages@inria.fr (François Fages), Sylvain.Soliman@inria.fr (Sylvain Soliman).

1 Introduction

Systems biology aims at elucidating the high-level functions of the cell from their biochemical basis at the molecular level [28]. A lot of work has been done for collecting genomic and post-genomic data and making them available in databases [1,29], and for organizing the knowledge on pathways and interaction networks into models of cell metabolism, signaling, cycle, apoptosis, etc. now published in model repositories (e.g. `http://biomodels.net/`). Furthermore the Systems Biology Markup Language (SBML) [27] provides a common exchange format for reaction models, which is nowadays supported by the majority of modeling tools [26,40].

Models of biological processes are built with two somewhat contradictory perspectives that are worth clarifying. The first perspective is a perspective of knowledge representation. In this perspective, the more concrete the better: models aim at gathering in a consistent way current knowledge on particular systems, and at representing the interactions participating in a system with the maximum of details. The second perspective for building models is to make predictions and answer particular questions about a system. Yet in this perspective, the more abstract the better: models for making predictions should get rid of useless details and should represent the minimum information that is sufficient for answering the questions at hand; the minimum the information the more powerful and efficient the tools available.

One way to reconcile these two perspectives is to put more focus on the issue of abstraction in systems biology, and to develop not only models but also their relationships to other models at different abstraction levels. In this paper we propose a formal ground for this issue by transposing the concepts of abstract interpretation and types borrowed from programming theory to systems biology. Abstract interpretation is a theory of abstraction, introduced by Cousot and Cousot in [15] as a framework for reasoning about programs, their semantics [14], and for designing static analyzers, among which type inference systems [13]. Type checking and type inference are important concepts and methods in programming languages and software engineering [5]. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyzes of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information.

In this paper, we investigate the application of these concepts to systems biology formalisms. More specifically, we consider the Systems Biology Markup Language SBML [27] and the Biochemical Abstract Machine BIOCHAM [4,20] with its differential, stochastic, discrete and boolean semantics [3,17]. We first

show how these different semantics can be formally related by simple Galois connections, as required in the theory of abstract interpretation, with the noticeable exception of the differential semantics that is discussed with some details.

Then we study three type systems:

(1) one for checking or inferring the protein functions in a reaction model,
(2) one for checking or inferring the activation and inhibition effects in a reaction model,
(3) and another one for checking or inferring the topology of compartments or locations in reaction models with space considerations.

We show that the framework of abstract interpretation elegantly applies to the formalization of these type abstractions, and to the implementation of linear or quadratic time complexity type checking as well as type inference algorithms. Furthermore, when comparing the inference of the activation and inhibition effects from the syntax of the reaction rules with their inference from the differential semantics, we show a theorem of independence of the graph of activation and inhibition effects from the kinetic expressions, under general conditions.

Through some examples of reaction models coming from the BioModels and BIOCHAM repositories [40], we show that the analysis of biochemical models by type inference provides accurate and useful information. Interestingly, we show that such a mathematical formalization of abstractions commonly used in systems biology already provides some guidelines for the extensions of biochemical reaction rule languages.

2 Preliminaries on Abstract Interpretation, Type Checking and Type Inference

2.1 Domains, Abstractions and Galois Connections

In the algebraic setting of abstract interpretation, a domain is a lattice $L(\sqsubseteq, \bot, \top, \sqcup, \sqcap)$ defined by a partial order $(L, \sqsubseteq)$, where $\bot$ and $\top$, elements of L and $\sqcup$, $\sqcap$, binary operators on L, respectively denote the least element, the greatest element, the least upper bound and the greatest lower bound. Intuitively, the partial ordering represents the information loss: the lesser the more informative, the greater the bigger loss of information.

As it is often the case in program analysis, the concrete domain and the

abstract domains considered for analyzing biochemical models, will be powersets, i.e. set lattices $\mathcal{P}(\mathcal{S})(\subseteq, \emptyset, \mathcal{S}, \cup, \cap)$ ordered by inclusion, with the empty set as $\bot$ element, and the base set $\mathcal{S}$ as $\top$ element. For instance, in the syntactical domain of reaction rule sets ordered by inclusion, the base set of all possible reactions makes all behaviors possible and thus contains no information, while the empty set is the most precise in this information ordering.

An *abstraction* is formalized by a Galois connection between a concrete domain $\mathcal{C}$ and an abstract domain $\mathcal{A}$, as follows [15]:

Definition 1 *A Galois connection* $\mathcal{C} \underset{\gamma}{\overset{\alpha}{\longleftrightarrow}} \mathcal{A}$ *between two lattices* $(\mathcal{C}, \sqsubseteq_C)$ *and* $(\mathcal{A}, \sqsubseteq_\mathcal{A})$ *is defined by an abstraction function* $\alpha : \mathcal{C} \to \mathcal{A}$*, and a concretization function* $\gamma : \mathcal{A} \to \mathcal{C}$*, that are monotonic:*

- $\forall\ c, c' \in \mathcal{C} : c \sqsubseteq_\mathcal{C} c' \ \Rightarrow\ \alpha(c) \sqsubseteq_\mathcal{A} \alpha(c')$,
- $\forall\ a, a' \in \mathcal{A} : a \sqsubseteq_\mathcal{A} a' \ \Rightarrow\ \gamma(a) \sqsubseteq_\mathcal{C} \gamma(a')$,

and are adjoint:

- $\forall c \in \mathcal{C}, \forall a \in \mathcal{A} : c \sqsubseteq_\mathcal{C} \gamma(a) \Leftrightarrow \alpha(c) \sqsubseteq_\mathcal{A} a$.

For any Galois connection, we have the following properties:

(1) $\gamma \circ \alpha$ is extensive (i.e. $c \sqsubseteq_\mathcal{C} \gamma \circ \alpha(c)$) and represents the information lost by the abstraction
(2) $\alpha \circ \gamma$ is contracting (i.e. $\alpha \circ \gamma(a) \sqsubseteq_\mathcal{A} a$)
(3) $\gamma \circ \alpha$ is the identity *iff* γ is onto *iff* α is one-to-one
(4) α preserves $\sqcup$, and γ preserves $\sqcap$
(5) $\gamma(a) = \max\ \alpha^{-1}(\downarrow a) = \sqcup \alpha^{-1}(\downarrow a)$
(6) $\alpha(c) = \min\ \gamma^{-1}(\uparrow c) = \sqcap \gamma^{-1}(\uparrow c)$
(7) the composition of two Galois connections is a Galois connection.

where $\downarrow a = \{b \mid b \sqsubseteq a\}$ and $\uparrow a = \{b \mid a \sqsubseteq b\}$.

If $\gamma \circ \alpha$ is the identity, the abstraction α loses no information, and $\mathcal{C}$ and $\mathcal{A}$ are isomorphic from the information standpoint (although α may be not onto and γ not one-to-one). It is equivalent in the definition of Galois connections to replace the condition of adjointness by conditions 1 and 2, or by condition 5 which also entails the monotonicity of γ.

Furthermore we shall use the fact that in powerset domains, the pointwise extension of any function from the base set of the concrete domain to the abstract domain forms a Galois connection:

Lemma 2 *Let* $\mathcal{C}$ *and* $\mathcal{A}$ *be two sets, and* $\alpha : \mathcal{P}(\mathcal{C}) \longrightarrow \mathcal{P}(\mathcal{A})$ *be a function*

such that $\alpha(c) = \bigcup_{e \in c} \alpha(\{e\})$. *Then the function* $\gamma(a) = \cup\alpha^{-1}(\downarrow a)$ *forms a Galois connection* $\mathcal{P}(\mathcal{C}) \underset{\gamma}{\overset{\alpha}{\longleftrightarrow}} \mathcal{P}(\mathcal{A})$ *between* $(\mathcal{P}(\mathcal{C}), \subseteq)$ *and* $(\mathcal{P}(\mathcal{A}), \subseteq)$.

Proof. We show that α is monotonic and $\gamma(a) = \max \alpha^{-1}(\downarrow a)$.

The monotonicity of α is immediate since if $c \subseteq c'$ we have $\bigcup_{c_i \in c} \alpha(\{c_i\}) \subseteq \bigcup_{c_i \in c'} \alpha(\{c_i\})$.

Now, let us consider $c = \gamma(a) = \cup\alpha^{-1}(\downarrow a)$, we need to prove that $c \in \alpha^{-1}(\downarrow a)$, i.e. $\alpha(c) \in \downarrow a$. We know that $\alpha(c) = \bigcup_{e \in c} \alpha(\{e\}) = \bigcup_{e \in \cup\alpha^{-1}(\downarrow a)} \alpha(\{e\})$. For each e in $\cup\alpha^{-1}(\downarrow a)$ there exists $d \in \mathcal{P}(\mathcal{C})$ such that $e \in d$ and $\alpha(d) \subseteq a$, therefore $\alpha(\{e\}) \subseteq a$. Hence $\bigcup_{e \in \cup\alpha^{-1}(\downarrow a)} \alpha(\{e\}) \subseteq a$ and thus $\alpha(c) \subseteq a$. □

In this paper, we will consider the syntactical domain of reaction models ordered by the inclusion of rule sets as concrete domain, and four semantical domains for respectively:

- the stochastic semantics, in which the reaction rules are interpreted by a continuous time Markov chain;
- the discrete semantics, in which the rules are interpreted by a Petri net;
- the boolean semantics, in which the rules are interpreted by a boolean asynchronous transition system;
- and the differential semantics, in which the rules are interpreted by a system of ordinary differential equations.

We will show in Sect. 3 that, with the noticeable exception of the differential semantics, all these domains are formally related by simple Galois connections.

2.2 Type Checking and Type Inference by Abstract Interpretation

Types provide further abstractions for reasoning about programs. In the setting of abstract interpretation, a type system $\mathcal{A}$ for a concrete domain $\mathcal{C}$ is nothing but a Galois connection $\mathcal{C} \underset{\gamma}{\overset{\alpha}{\longleftrightarrow}} \mathcal{A}$. The *type inference* problem is, given a concrete element $x \in \mathcal{C}$ (e.g. a reaction model), to compute $\alpha(x)$ (e.g. the protein functions that can be inferred from the reactions). The *type checking* problem is, given a concrete element $x \in \mathcal{C}$ and a typing $y \in \mathcal{A}$ (e.g. a set of protein functions), to determine whether $x \sqsubseteq_{\mathcal{C}} \gamma(y)$ (i.e. whether the reactions are compatible with the information given on the protein functions) which is equivalent to $\alpha(x) \sqsubseteq_{\mathcal{A}} y$ (i.e. whether the given typing contains the inferred types).

Most of the type systems considered in this paper will be implemented with

type checking and type inference algorithms that basically browse the set of reactions, and check or collect the type information for each rule or pair of rules independently, thus in linear time or quadratic time respectively.

In this paper, we will consider three abstract domains for types:

- one for protein functions, where molecules are abstracted into categories such as kinases and phosphatases (Sect. 4),
- one for the influence graph, where the biochemical reaction rules are abstracted by binary relations of activation and inhibition between molecular species (Sect. 5),
- and one for location topologies, where reaction and transport rules are abstracted by retaining only the neighborhood information between locations (Sect. 6).

These domains will be defined by abstractions from the syntactical domain of reaction models. The syntactical domain indeed suffices to define the abstractions necessary for these analyses. It is worth noting that a similar situation also occurs in program analysis when the syntax of programs captures enough of the semantics for the needs of the analysis. For the analysis of influences between species, we will compare in section 5 the results obtained by abstraction from the syntactical domain, with the information obtained by abstraction from the differential semantics.

3 Domains for Reaction Models and Hierarchy of Semantics

3.1 Syntactical Domain of Reaction Models

Following SBML and BIOCHAM conventions, a model of a biochemical system is a set of reaction rules of the form e `for` l `=>` r where l is a multiset of molecule names given with stoichiometric coefficients, called a *solution*, r is the transformed solution, and e is a *kinetic expression*, i.e. a *positive* arithmetic expression on the concentrations of the molecules in i (plus possibly of some other molecules that have for instance an inhibitory effect on the reaction).

We will use the BIOCHAM operators `+` and `*` to denote solutions as `2*A + B`, as well as the syntax of catalyzed reactions `e for 2*A+B =[C]=> D` as an abbreviation for `e for 2*A+B+C => C+D`. By abuse of notation, assuming a finite set of molecules $\mathcal{M}$, we shall also see a solution l as an $|\mathcal{M}|$-dimensional vector of integers, and will denote by $l(A)$ the stoichiometric coefficient of A in solution l.

Formally, the concrete domain of reaction models is the powerset of all possible reaction rules ordered by set inclusion :

Definition 3 *Given a finite set $\mathcal{M}$ of molecule names, the universe of reactions is the set of rules*

$$\mathcal{R} = \{e \texttt{ for } l \texttt{ => } r \mid \quad e \text{ is a kinetic expression,} \\ \text{and } l \text{ and } r \text{ are solutions of molecules in } \mathcal{M}\}.$$

The concrete domain $\mathcal{D}_{\mathcal{R}} = (\mathcal{P}(\mathcal{R}), \subseteq)$ of reaction models *is the power-set of reaction rules ordered by inclusion.*

Note that in this domain, the composition of two reaction models is naturally the union of the sets of reactions. A reaction appearing in two reaction sets is thus not duplicated when composing two models by set union.

In the SBML exchange format, no particular semantics is defined, and this syntactical domain is the natural one to consider. In BIOCHAM, reaction models are interpreted under four semantics that correspond to four different abstraction levels : the boolean semantics, the discrete semantics, the differential semantics and the stochastic semantics [3,17]. In the following subsections, we formalize these semantical domains and study their formal relationship by Galois connections within a hierarchy of semantics.

It is worth noting that in the context of programming languages, it is not usual (and generally not possible) to include the syntactical domain ordered y set inclusion within the hierarchy of semantics. It is possible here however for the rule-based language of reactions, and should be possible as well for other rule-based languages in which programs can be ordered by set inclusion, like Prolog for instance [12].

3.2 Stochastic Semantics

The most realistic interpretation of biochemical reaction models is provided by the *stochastic semantics*. In that semantics, a reaction model is interpreted as a (continuous time) Markov chain, and the kinetic expressions as transition rates. This interpretation is correct w.r.t. the Master Chemical Equation if we suppose that the reactions happen in a well stirred environment (i.e. "instantaneous" diffusion) with constant pressure, temperature and volume [25].

For a given volume V_k of the location where the compound x_k resides, a concentration C_k for x_k is translated into a molecule number $N_k = \lfloor C_k \times V_k \times N_A \rfloor$,

where N_A is Avogadro's number. A state in the stochastic semantics will be a vector of integers indicating the numbers of molecules for each species.

Formally, given a fixed finite set $\mathcal{M}$ of molecule names, the stochastic transition semantics is defined by the following domain :

Definition 4 *Let a* discrete state *be a vector of positive integers of dimension* $|\mathcal{M}|$. *The universe* $\mathcal{S}$ *of* stochastic transitions *is the set of triplets* (S, S', τ) *where* S *and* S' *are discrete states and* $\tau \in \mathbb{R}^+$ *is a weight. The domain* $\mathcal{D}_{\mathcal{S}} = (\mathcal{P}(\mathcal{S}), \subseteq)$ *of* stochastic transition models *is the power-set of stochastic transitions ordered by inclusion.*

Note first that discrete states have the same mathematical structure as solutions in reaction rules, and can both be represented by $|\mathcal{M}|$-dimensional vectors of positive integers. In the following, we will identify states and solutions and will sum them (see definition of $S \rightarrow_i S'$ below and theorem 13).

Note also that in a stochastic transition model s, there can be more than one transition from one state to another one, labelled with different real numbers. We define the *weight* in s of a transition from state S_i to S_j as the sum of the weights $\tau_{ij} = \sum_{(S_i,S_j,\tau)\in s} \tau$.

Now, an element s of the domain precisely defines a Markov chain where the *probability* p_{ij} of having a transition from state S_i to state S_j is obtained by normalizing the transition weights into $p_{ij} = \frac{\tau_{ij}}{\sum_k \tau_{ik}}$. Then the transition time can be computed as usual. Stochastic simulation techniques like Gillespie's algorithm [24] compute realizations of the processes described by models in the stochastic domain, where random variables range over the probability and the time of transition. The results of those simulations are generally noisy versions of the simulation obtained by the interpretation of the reaction rules by a system of ordinary differential equations (see section 3.5). However, in models with for instance, very few molecules of some kind, qualitatively different behaviors may appear in the stochastic simulation, and thus justify the recourse to that semantics in such cases. A classical example is the model of the lambda phage virus [22] in which a small number of molecules, promotion factors of two genes, can generate an explosive multiplication (lysis) after a more or less long period of passive wait (lysogeny).

Now, in order to relate the stochastic semantics domain to the syntactical domain of reaction rules, let us consider a reaction rule model $\{e_i \texttt{ for } l_i \texttt{=>} r_i\}_{i\in I}$, and denote by $S \rightarrow_i S'$ the fact that *rule i fires in state* S resulting in state S', i.e. if $S \geq l_i$ (pointwise) and $S' = S - l_i + r_i$.

In a given state S, the numbers of molecules are fixed integer values and the kinetic expression e_i evaluates into a (positive) real valued *reaction rate*, noted $e_i(S)$. This allows us to relate the stochastic transition domain to the

syntactical domain of reaction rules by the following Galois connection :

Proposition 5 *Let $\alpha_{\mathcal{RS}} : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{S}}$ be the function associating to a reaction model $\{e_i$ `for` l_i`=>`$r_i\}_{i\in I}$ the stochastic transition model $\{(S, S', e_i(S)) \in \mathcal{S} \mid i \in I,\ S \to_i S'\}$. Let $\gamma_{\mathcal{RS}}(s) = \cup{\alpha_{\mathcal{RS}}}^{-1}(\downarrow s)$. $\mathcal{D}_{\mathcal{R}} \underset{\gamma_{\mathcal{RS}}}{\overset{\alpha_{\mathcal{RS}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{S}}$ is a Galois connection.*

Proof. Simply note that $\alpha_{\mathcal{RS}}$ is defined by its union on each rule of the concrete model and apply Lemma 2. □

Proposition 6 *$\alpha_{\mathcal{RS}}$ is not one-to-one.*

Proof. For instance, the reaction models $m1 =$ { `e for A => B`} and $m2 = m1 \cup$ { `e for 2*A => A+B`} have the same set of stochastic transitions. $\gamma \circ \alpha$ is thus not the identity, the information lost by the stochastic abstraction is the elimination of redundant rules in the reaction model. □

$\alpha_{\mathcal{RS}}$ is neither onto as the stochastic transitions obtained from a reaction model enjoy some particular properties, such as for instance the following stability property w.r.t. the number of molecules in the states:

Proposition 7 *If two states S_1, S_2 are such that $S_1 \leq S_2$ pointwise, then for any reaction model m and any stochastic transition $(S_1, S, \tau) \in \alpha_{\mathcal{RS}}(m)$, we have $(S_2, (S + S_2 - S_1), \tau) \in \alpha_{\mathcal{RS}}(m)$, i.e. all the rules that apply in S_1 apply in S_2 with the same effect.*

Proof. By definition of $\alpha_{\mathcal{RS}}$. □

Corollary 8 *$\alpha_{\mathcal{RS}}$ is not onto.*

3.3 Discrete Semantics

The discrete semantics of reaction models can be defined as the trivial abstraction of the stochastic semantics that simply forgets the transition rates.

Definition 9 *The universe $\mathcal{D}$ of* discrete transitions *is the set of pairs of discrete states. The domain $\mathcal{D}_{\mathcal{D}}$ of discrete transitions is the power-set of discrete transitions ordered by inclusion $\mathcal{D}_{\mathcal{D}} = (\mathcal{P}(\mathcal{D}), \subseteq)$.*

Proposition 10 *Let $\alpha_{\mathcal{SD}} : \mathcal{D}_{\mathcal{S}} \to \mathcal{D}_{\mathcal{D}}$ be the function associating to a set of stochastic transitions the discrete transitions obtained by projection on the two first components, and $\gamma_{\mathcal{SD}}(d) = \cup{\alpha_{\mathcal{SD}}}^{-1}(\downarrow d)$. $\mathcal{D}_{\mathcal{S}} \underset{\gamma_{\mathcal{SD}}}{\overset{\alpha_{\mathcal{SD}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{D}}$ is a Galois connection.*

Proof. Here again it suffices to note that $\alpha_{\mathcal{SD}}$ is defined by its union on each single stochastic transition of the concrete model and to apply Lemma 2. □

Remark that $\alpha_{\mathcal{SD}}$ is this time onto, but obviously not one-to-one as the transition rates are simply forgotten.

It is worth noticing that the discrete semantics corresponds to the classical Petri net semantics of reaction models [36,37,9,23]. As a consequence, classical Petri net analysis tools can be used for the analysis of reaction models at this abstraction level. For instance, the elementary mode analysis of metabolic networks [38] has been shown in [45] to be equivalent to the classical analysis of Petri nets by T-invariants. These analyses apply to the discrete semantics of reaction models in all generality.

3.4 Boolean Semantics

The boolean semantics is purely qualitative, and provides somehow the most abstract semantics of reaction models. The boolean semantics forgets the kinetic expressions and interprets the rules as a (non-deterministic) asynchronous transition system but this time over boolean states representing the absence or presence of molecules. It can be applied to large models for which the kinetic data may be not available.

Definition 11 *Let a* boolean state *be a vector of booleans of dimension* $|\mathcal{M}|$ *indicating the presence of each molecule in the state. The universe* $\mathcal{B}$ *of* boolean transitions *is the set of pairs of boolean states. The domain* $\mathcal{D}_{\mathcal{B}}$ *of boolean transitions is the power-set of boolean transitions ordered by inclusion* $\mathcal{D}_{\mathcal{B}} = (\mathcal{P}(\mathcal{B}), \subseteq)$.

This semantical domain is related to the discrete transitions semantics domain by the zero/non-zero abstraction from the integers to the booleans, and its pointwise extension from discrete states to boolean states $\alpha_{\mathcal{NB}} : \mathbb{N}^{|\mathcal{M}|} \to \mathbb{B}^{|\mathcal{M}|}$.

Proposition 12 *Let* $\alpha_{\mathcal{DB}} : \mathcal{D}_{\mathcal{D}} \to \mathcal{D}_{\mathcal{B}}$ *be the function associating to a set of discrete transitions the set of boolean transitions obtained by applying* $\alpha_{\mathcal{NB}}$ *to the discrete states. Let* $\gamma_{\mathcal{DB}}(b) = \cup {\alpha_{\mathcal{DB}}}^{-1}(\downarrow b)$. $\mathcal{D}_{\mathcal{D}} \underset{\gamma_{\mathcal{DB}}}{\overset{\alpha_{\mathcal{DB}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{B}}$ *is a Galois connection.*

Proof. As before, note that $\alpha_{\mathcal{DB}}$ is defined by its union on each transition of the concrete model and apply Lemma 2. □

In BIOCHAM, the boolean semantics of reaction models is computed by associating to each reaction rule a set of boolean transition rules that take into

account the possible complete consumption or not of the reactants by the reaction [7]. For instance, a reaction rule like `A+B=>C+D` is interpreted by four boolean transition rules :

- $A \wedge B \longrightarrow A \wedge B \wedge C \wedge D$
- $A \wedge B \longrightarrow \neg A \wedge B \wedge C \wedge D$
- $A \wedge B \longrightarrow A \wedge \neg B \wedge C \wedge D$
- $A \wedge B \longrightarrow \neg A \wedge \neg B \wedge C \wedge D$

Given a reaction model R, let us denote by S_{BB} the set of boolean transitions obtained by applying these boolean transition rules to each state. The following theorem shows that the BIOCHAM boolean semantics of reaction models *over-approximates* the boolean semantics obtained from the quantitative semantics. The non-existence of a behaviour in the BIOCHAM boolean semantics thus entails its non-existence in the quantitative semantics of the rules whatever the kinetic expressions are.

Theorem 13 *For any reaction model R, $\alpha_{\mathcal{DB}}(\alpha_{\mathcal{SD}}(\alpha_{\mathcal{RS}}(R))) \subseteq S_{BB}$.*

Proof. Since all our abstractions are defined pointwise, it is enough to prove it for only one rule in R. Let us consider e `for` l `=>` r. By abuse of notation we will denote by l and r the discrete states corresponding to solutions of same name. We have $\alpha_{\mathcal{RS}}(R) = \{(S_i, S_j, e) | S_i \geq l,\ S_j = S_i - l + r\}$ and thus $\alpha_{\mathcal{SD}}(\alpha_{\mathcal{RS}}(R)) = \{(S_i, S_j) | S_i \geq l,\ S_j = S_i - l + r\}$, which leads to $\alpha_{\mathcal{DB}}(\alpha_{\mathcal{SD}}(\alpha_{\mathcal{RS}}(R))) = \{(S'_i, S'_j) | S_i \geq l,\ S_j = S_i - l + r,\ S'_i = \alpha_{\mathcal{NB}}(S_i),\ S'_j = \alpha_{\mathcal{NB}}(S_j)\}$. Since $S_{BB} = \{(T, T') | T \geq \alpha_{\mathcal{NB}}(l),\ \alpha_{\mathcal{NB}}(r) \vee (T \wedge \neg\alpha_{\mathcal{NB}}(l)) \leq T' \leq \alpha_{\mathcal{NB}}(T) \vee \alpha_{\mathcal{NB}}(r)\}$ we can see that the property holds as $S_i \geq l$ implies $S'_i \geq \alpha_{\mathcal{NB}}(l)$, and since $S_i \geq l$ we have $S_j = S_i - l + r \Rightarrow S_i - l + r \leq S_j \leq S_i + r \Rightarrow \alpha_{\mathcal{NB}}(S_i - l + r) = \alpha_{\mathcal{NB}}(r) \vee (\alpha_{\mathcal{NB}}(S_i) \wedge \neg\alpha_{\mathcal{NB}}(l)) \leq S'_j \leq \alpha_{\mathcal{NB}}(S_i + r) = \alpha_{\mathcal{NB}}(S_i) \vee \alpha_{\mathcal{NB}}(r)$. □

It is worth noticing that this property does not hold for the boolean semantics of reaction models that always assume either incomplete consumption, or complete consumption, like in Pathway Logic [16] or in boolean Petri nets [23]. In these formalisms, the correctness of the boolean interpretation w.r.t. a quantitative interpretation is thus left to the modeler who is in charge of explicitly adding reaction rules for the different cases of consumption of the reactants.

3.5 Differential Semantics

The *differential semantics* of reaction models interprets a set of reaction rules $\{e_i$ `for` l_i `=>` $r_i\}_{i=1,...,n}$ over molecular concentration variables $\{x_1, ..., x_m\}$, by

the following system of Ordinary Differential Equations (ODE):

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

where $r_i(x_k)$ (resp. l_i) is the stoichiometric coefficient of x_k in the right (resp. left) member of rule i. Thanks to its wide range of mathematical tools, this semantics is the most commonly used in mathematical biology [39].

The study of the relationship between the differential and the stochastic semantics dates back to the seminal work of Boltzmann in the XIXth century who created the domain of statistical physics. In this setting, the differential semantics is obtained from the stochastic semantics by limit operations where the number of molecules tends to the infinity and the time steps tend to zero. under several assumptions such as perfect diffusion.

In the setting of abstract interpretation, the differential semantics is however difficult to formally relate to the previous semantics for several reasons. The differential semantics is a synchronous semantics in the sense that it specifies the evolution of variables in parallel, whereas all the other semantics are asynchronous in the sense that the interleaving semantics is considered where one reaction is fired at a time. Hence the notion of time is not the same in both categories of semantics, having infinitesimal time steps in the differential semantics, and time for one transition in the other semantics. Furthermore the differential semantics is deterministic and produces an average trace, whereas the other semantics produce sets of possible traces representing the competition between reactions.

For these reasons, the differential semantics does not belong to our hierarchy of syntactical, stochastic, discrete and boolean semantics. In section 5, we will come back to it however for comparing the analysis of the influence graph between molecules obtained from the differential semantics, with the one obtained from the syntax of the reaction rules, and for establishing equivalence results under some general conditions on the kinetics.

4 A Type System for Protein Functions

In this section, we investigate the use of types for formally relating information on the biological *function* of some proteins to reaction models. For the sake of simplicity, we restrict ourselves to two simple enzymatic functions: kinase and phosphatase. These functions correspond to the action of adding (resp. removing) a phosphate group to (resp. from) a compound with a covalent binding. We do not consider other categories such as protease in degradation

rules, nor acetylase and deacetylase in modification rules, etc. This choice is in accordance with the BIOCHAM syntax which permits to mark the sites of a protein where a group is added, with the operator `~`, as in `P~{p,q}` where protein `P` is modified on its sites `p` and `q`, without distinguishing however between phosphorylation, acetylation, methylation, ubiquitination, etc. We thus consider BIOCHAM models containing compounds with different levels of phosphorylation or acetylation, etc. without distinguishing the different forms of modification, and call them phosphorylation by abuse of terminology.

The inference of protein functions in a reaction model is interesting for several reasons. First, the kind of information (kinase activity) collected on proteins can be checked using online databases like for instance GO, the Gene Ontology [1]. Second, in the context of the machine learning techniques implemented in BIOCHAM for completing or revising a model w.r.t. a temporal logic specification [3], the information that an enzyme acts as a kinase or as a phosphatase drastically reduces the search space for adding reactions, and helps to directly find rules and model revisions that are biologically plausible.

4.1 Abstract Domain of Protein Functions

Definition 14 *Let kinase(A,B) and phosphatase(A,B) be relations in* $\mathcal{M} \times \mathcal{M}$ *denoting the kinase (resp. phosphatase) function of A on B. The abstract domain of* protein functions $\mathcal{D}_{\mathcal{F}} = \mathcal{P}(\{kinase(A,B) \mid A, B \in \mathcal{M}\} \cup \{phosphatase(A,B) \mid A, B \in \mathcal{M}\})$ *is the powerset of these expressions, ordered by inclusion.*

The abstraction function from the syntactical domain, $\alpha_{\mathcal{F}} : \mathcal{D}_{\mathcal{R}} \rightarrow \mathcal{D}_{\mathcal{F}}$*, associates to a reaction model R the union of the abstractions defined for each single rule and each pair of rules as follows:*

$\alpha_{\mathcal{F}}$(`A =[B]=> C`) = {*kinase(B,A)*} *if* `C` *is more phosphorylated than* `A` *(i.e. its set of active phosphorylation sites strictly includes that of* `A`*), in which case* B has kinase function w.r.t. A*;*

$\alpha_{\mathcal{F}}$(`A + B => D, D => C + B`) = {*kinase(B,A)*} *if similarly* `C` *is more phosphorylated than* `A`*;*

$\alpha_{\mathcal{F}}$(`A =[B]=> C`) = {*phosphatase(B,A)*} *if, on the contrary,* `A` *is more phosphorylated than* `C`*;*

$\alpha_{\mathcal{F}}$(`A + B => D, D => C + B`) = {*phosphatase(B,A)*} *if* `A` *is more phosphorylated than* `C`*.*

Note that as the abstraction function is not defined pointwise but also on

pairs of reaction rules, the time complexity for computing the set of protein functions from the reactions is quadratic in the number of rules. One can easily check that:

Proposition 15 *Let* $\gamma_{\mathcal{F}}(f) = \cup {\alpha_{\mathcal{F}}}^{-1}(\downarrow f)$, $\mathcal{D}_{\mathcal{R}} \underset{\gamma_{\mathcal{F}}}{\overset{\alpha_{\mathcal{F}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{F}}$ *is a Galois connection.*

This typing for protein functions is very precise as it refers to particular molecules. On the other hand, keeping only the kinase or phosphatase function in an unary predicate without the information on the transformed molecules might be too loose. Between these two extreme choices, one could also consider a hierarchical type structure such as the one defined by the following grammar:

$$\tau ::= kinase|phosphatase|kinase(\tau)|phosphatase(\tau)|T$$

where T denotes some basic types of proteins, with the subtyping relations $kinase(\tau) \preceq kinase$ and $phosphatase(\tau) \preceq phosphotase$. This kind of typing relation stems from models like the MAPK cascade shown in next section where the common denomination for the function of MEK is "MAPK kinase" (i.e. $kinase(MAPK)$) and that of RAF is "MAPK kinase kinase" (i.e. $kinase(kinase(MAPK))$). It is worth noting that such typings are supported by type systems already defined for rule based languages as in [18], using solvers for subtyping constraints in general ordering structures such as quasi-lattices [11] for instance. These considerations are however beyond the scope of this paper and will not be further developed here.

4.2 Evaluation Results

4.2.1 MAPK model.

On a simple example of the MAPK cascade originally based on [32] and imported into BIOCHAM, the type inference algorithm determines that `RAFK`, `RAF~{p1}` and `MEK~{p1,p2}` have a kinase function; `RAFPH`, `MEKPH` and `MAPKPH` have a phosphatase function; and the other compounds have no function inferred.

If the family of MAPK molecules is given as a basic type, one would moreover infer that the active form of `MEK` is a MAPKK (a kinase for the MAPK family), and that the active form of `RAF` is a MAPKKK (a MAPKK kinase).

If we wanted to type-check such a model, we would correctly check all phosphatases but would miss an example of the kinase function of `MAPK~{p1,p2}`, since its action is not visible in the above model.

4.2.2 Kohn's Map.

Kohn's map of the mammalian cell cycle control [31] has been transcribed in BIOCHAM to serve as a large benchmarking example of 500 species and 800 rules [8]. This example shows that this abstraction scales up efficiently as the computation of influences requires less than one second CPU time (on a PC 1,7GHz) in this model. Here is an excerpt of the output of the type inference, where it was restricted to the unary functions *kinase* and *phosphatase* as explained at the end of section 4.1:

```
cdk7-cycH is a kinase
Wee1 is a kinase
Myt1 is a kinase
cdc25C~{p1} is a phosphatase
cdc25C~{p1,p2} is a phosphatase
Chk1 is a kinase
C-TAK1 is a kinase
Raf1 is a kinase
cdc25A~{p1} is a phosphatase
cycA-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycE-cdk2~{p2} is a kinase
cdk2~{p2}-cycE~{p1} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
cycA-cdk1~{p3} is a kinase
cycB-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
Plk1 is a kinase
pCAF is a kinase
p300 is a kinase
HDAC1 is a phosphatase
```

On the other hand in these results, no compound is both a kinase and a phosphatase. The protein `cdc25A`, `cdc25C` and `HDAC1` are the only phosphatases found in the whole map. The type inference also tells us that the cyclin-dependant kinases have a kinase function when in complex with a cyclin, which is correct. Finally the acetylases `pCAF`, `p300` and the deacetylase `HDAC1` are detected but as expected identified to kinases and phosphatases respectively, since the BIOCHAM syntax does not distinguish between phosphorylation and acetylation.

5 A Type System for Activation and Inhibitory Influences

5.1 Abstract Domain of Influences

Influence networks for activation and inhibition have been introduced for the analysis of gene expression in the setting of gene regulatory networks [42], they basically define graphs where vertices are genes and oriented edges are labelled either with *activates* or *inhibits*, representing the supposed regulation of one gene by another one. Such influence networks are in fact an abstraction of complex reaction networks, and can be applied as such to protein interaction networks. However the distinction between the influence network and the reaction network is crucial for the application of Thomas's conditions of multistationarity and oscillations [42,41] to protein interaction networks, and there has been some confusion between the two kinds of networks [34]. Here we precisely define influence networks as an abstraction (a type system) of reaction networks.

Definition 16 *The abstract domain of influences is the powerset of the binary relations of activation and inhibition between compounds* $\mathcal{D}_{\mathcal{I}} = \mathcal{P}(\{A \text{ activates } B \mid A, B \in \mathcal{M}\} \cup \{A \text{ inhibits } B \mid A, B \in \mathcal{M}\})$*, ordered by inclusion.*

5.2 Abstraction from the syntax of the reaction rules

Definition 17 *The influence abstraction* $\alpha_{\mathcal{RI}} : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{I}}$ *is the function*

$$\begin{aligned} \alpha_{\mathcal{RI}}(x) = \quad & \{A \text{ activates } B \quad \mid \exists (e_i \text{ for } l_i \Rightarrow r_i) \in x, \\ & \qquad\qquad\quad\; l_i(A) > 0 \text{ and } r_i(B) - l_i(B) > 0\} \\ & \cup \{A \text{ inhibits } B \quad \mid \exists (e_i \text{ for } l_i \Rightarrow r_i) \in x, \\ & \qquad\qquad\quad\; l_i(A) > 0 \text{ and } r_i(B) - l_i(B) < 0\} \end{aligned}$$

In particular, we have the following influences for elementary reactions of complexation, modification, synthesis and degradation:

$$\alpha_{\mathcal{RI}}(\{\texttt{A} + \texttt{B} \texttt{=>} \texttt{C}\}) = \{ \quad \text{A inhibits B, A inhibits A, B inhibits A,}$$
$$\text{B inhibits B, A activates C, B activates C}\}$$
$$\alpha_{\mathcal{RI}}(\{\texttt{A} = [\texttt{C}] \texttt{=>} \texttt{B}\}) = \{ \quad \text{C inhibits A, A inhibits A,}$$
$$\text{A activates B, C activates B}\}$$
$$\alpha_{\mathcal{RI}}(\{\texttt{A} = [\texttt{B}] \texttt{=>} \_\}) = \{ \quad \text{B inhibits A, A inhibits A}\}$$
$$\alpha_{\mathcal{RI}}(\{\_ = [\texttt{B}] \texttt{=>} \texttt{A}\}) = \{ \quad \text{B activates A}\}$$

The inhibition loops on the reactants are justified by the negative sign in the differential semantics of the reactions (see theorem 21 of the next section). These loops are however often omitted in the influence graphs considered in the literature, together with some other influences, according to functionality, kinetics and non-linearity considerations [30].

The abstraction function $\alpha_{\mathcal{RI}}$ allows us either to type check a reaction model w.r.t. a given influence typing of molecules, or to infer the influence types from the reaction rules. As $\alpha_{\mathcal{RI}}$ is defined pointwise, it can be computed very efficiently in linear time, and we have by lemma 2:

Proposition 18 *Let* $\gamma_{\mathcal{RI}}(f) = \cup{\alpha_{\mathcal{RI}}}^{-1}(\downarrow f)$, $\mathcal{D}_{\mathcal{R}} \underset{\gamma_{\mathcal{RI}}}{\overset{\alpha_{\mathcal{RI}}}{\longleftrightarrow}} \mathcal{D}_{\mathcal{I}}$ *is a Galois connection.*

5.3 Abstraction from the differential semantics of reaction rules

In the differential semantics of a reaction rule model $\{e_i \texttt{ for } l_i\texttt{=>}r_i \mid i \in I\}$ we have $\dot{x_k} = dx_k/dt = \sum_{i=1}^{n}(r_i(x_k) - l_i(x_k)) * e_i$. The Jacobian matrix J is formed of the partial derivatives $J_{ij} = \partial\dot{x_i}/\partial x_j$, and one can define the domain $\mathcal{D}_{\mathcal{J}}$ of Jacobians ordered by the pointwise inclusion of codomains. Let us denote by β the mapping from $\mathcal{D}_{\mathcal{R}}$ to $\mathcal{D}_{\mathcal{J}}$ that extracts $\dot{x_k}$ (by the equation given at the beginning of this paragraph) and hence the Jacobian from the kinetic expressions in the reaction rules.

Definition 19 *The differential influence abstraction* $\alpha_{\mathcal{JI}} : \mathcal{D}_{\mathcal{J}} \to \mathcal{D}_{\mathcal{I}}$ *is the function*

$$\alpha_{\mathcal{JI}}(x) = \{A \textit{ activates } B \mid \partial\dot{x_B}/\partial x_A > 0 \textit{ in some point of the space}\}$$
$$\cup\{A \textit{ inhibits } B \mid \partial\dot{x_B}/\partial x_A < 0 \textit{ in some point of the space}\}$$

The comparison between the differential influences, represented by the function $\alpha_{\mathcal{JI}} \circ \beta$, and the syntactical influences, represented by the abstraction function $\alpha_{\mathcal{RI}}$, requires that the information in the kinetic expressions and

in the reaction rules are compatible. This motivates the following definition where, intuitively, the first property forbids the absence of purely kinetic inhibitors not represented in the rules, and the second property enforces that reactants and enzymes do appear in rules where they are used.

Definition 20 *In a reaction model* $x = \{e_i \texttt{ for } l_i \texttt{=>} r_i \mid i \in I\}$*, we say that a kinetic expression* e_i *is* monotonic *iff for all molecules* x_k *we have*

(1) $\partial e_i/\partial x_k \geq 0$ *in all points of the space,*
(2) $l_i(x_k) > 0$ *whenever* $\partial e_i/\partial x_k > 0$ *in some point of the space.*

A reaction model x *has a* monotonic kinetics *iff all its reaction rules have monotonic kinetics.*

Note that the mass action law kinetics, $e_i = k * \Pi {x_i}^{l_i}$, are monotonic and that Hill's kinetics (of which Michaelis-Menten kinetics are a special case with $n = 1$) $e_i = V_m * {x_s}^n/(K_m + {x_s}^n)$ where $V_m = k * (x_e + x_e * x_s/K_m)$ for an enzymatic reaction $x_s = [x_e] => x_p$, are also monotonic [1] . On the other hand, inhibitions with negative Hill kinetics of the form $e_i = V_m/(K_m + {x_s}^n)$ are not monotonic, and are not reflected in the syntax of the reactants of the rules.

Theorem 21 *For any reaction model* x *with monotonic kinetics,* $\alpha_{\mathcal{JI}} \circ \beta(x) \subseteq \alpha_{\mathcal{RI}}(x)$.

Proof. If (A activates B) $\in \alpha_{\mathcal{JI}} \circ \beta(x)$ then $\partial \dot{B}/\partial A > 0$. Hence there exists a term in the differential semantics, of the form $(r_i(B) - l_i(B)) * e_i$ with $\partial e_i/\partial A$ of the same sign as $r_i(B) - l_i(B)$.

Let us suppose that $r_i(B) - l_i(B) > 0$ then $\partial e_i/\partial A > 0$ and since e_i is monotonic we get that $l_i(A) > 0$ and thus that (A activates B) $\in \alpha_{\mathcal{RI}}(x)$. If on the contrary $r_i(B) - l_i(B) < 0$ then $\partial e_i/\partial A < 0$, which is not possible for a monotonic kinetics.

If (A inhibits B) $\in \alpha_{\mathcal{JI}} \circ \beta(x)$ then $\partial \dot{B}/\partial A < 0$. Hence there exists a term in the differential semantics, of the form $(r_i(B) - l_i(B)) * e_i$ with $\partial e_i/\partial A$ of sign opposite to that of $r_i(B) - l_i(B)$.

Let us suppose that $r_i(B) - l_i(B) > 0$ then $\partial e_i/\partial A < 0$, which is not possible for a monotonic kinetics. If on the contrary $r_i(B) - l_i(B) < 0$ then $\partial e_i/\partial A > 0$ and since e_i is monotonic we get that $l_i(A) > 0$ and thus that (A activates B) $\in \alpha_{\mathcal{RI}}(x)$. □

[1] $x_e * x_s/K_m$ is the concentration of the enzyme-substrate complex, supposed constant in the Michaelian approximation and $x_e + x_e * x_s/K_m$ is thus the total amount of enzyme

It is worth noticing that even in the simple case of mass action law kinetics, there is no equality between $\alpha_{\mathcal{JI}} \circ \beta$ and $\alpha_{\mathcal{RI}}$. For instance let x be the following model :

$k_1 * A$ for $A => B$

$k_2 * A$ for $\_ = [A] => A$

We have $\alpha_{\mathcal{RI}}(x) = \{$A activates B, A activates A, A inhibits A$\}$, however $\dot{A} = (k_2 - k_1) * A$, hence $\partial \dot{A} / \partial A$ can be made always positive or always negative or always null, resulting in the absence from $\alpha_{\mathcal{JI}} \circ \beta(x)$ of, respectively, A inhibits A, A activates A or both.

Actually in the general case, β is not monotonic since adding rules can compensate an existing rule in the differential expression and eliminate terms in the differential equations. The differential semantics is thus not an abstraction of the reaction models ordered by set inclusion in the formal sense of abstract interpretation. The above case shows that $\alpha_{\mathcal{JI}} \circ \beta$ applied to the first rule contains *A inhibits A*, whereas its application to the set of two rules (greater in $\mathcal{D}_{\mathcal{R}}$) may not. A sufficient condition for β to be monotonic is that in the model no kinetic expression can compensate another one in the Jacobian. That is : $\forall x_i, x_j \exists ?k$ s.t. $r_k(x_i) \neq l_k(x_i)$ and $\partial e_k / \partial x_j \neq 0$. This condition is used in the forthcoming corollary 25. Furthermore, under some hypotheses about the adequateness between the kinetic expressions and the rules, shown to be quite general in the following section, the equality holds between both abstractions.

Definition 22 *In a reaction model* $x = \{e_i$ `for` l_i`=>`$r_i \mid i \in I\}$*, a kinetic expression* e_i *is* strongly monotonic *iff for all molecules* x_k *we have*

(1) $\partial e_i / \partial x_k \geq 0$ *in all points of the space,*
(2) $l_i(x_k) > 0$ *iff there exists a point in the space s.t.* $\partial e_i / \partial x_k > 0$

A reaction model x *has a* strongly monotonic kinetics *iff all its reaction rules have a strongly monotonic kinetics.*

Note that *strongly monotonic* implies *monotonic.*

Proposition 23 *Mass action law, Michaelis Menten, and Hill kinetics are strongly monotonic.*

Proof.

For the case of Mass action law, the kinetics are of the form:

$$e_i = k_i \prod_{l=1}^{m} x_l^{l_i(x_l)}$$

with $k_i > 0$ and $l_i(x_l) \geq 0$. We thus have $\partial e_i/\partial x_k = 0$ if $l_i(x_k) = 0$ and $\partial e_i/\partial x_k = k_i * l_i(x_k) * x_k^{l_i(x_k)-1} \prod_{l \neq k} x_l^{l_i(x_l)}$ otherwise, which clearly satisfies (1) and (2).

In the case of Hill kinetics (of which Michaelis Menten is a subcase), we have:

$$e_i = \frac{V_m * x_s^n}{K_m^n + x_s^n}$$

for the reaction $x_s + x_e => x_p + x_e$ and where $V_m = k_2 * x_e^{tot} = k_2 * (x_e + k_1 * x_e * x_s/(k_{-1} + k_2))$ from the steady state approximation. It is obvious that $\partial e_i/\partial x_k = 0$ for all x_k other than x_s and x_e since they do not appear in e_i and one can easily check that with all the constants n, k_1, k_{-1}, k_2 strictly positive, both $\partial e_i/\partial x_e$ and $\partial e_i/\partial x_s$ are greater than 0 at some point in the space.

□

Lemma 24 *Let x be a reaction model with strongly monotonic kinetics, and A and B be two molecules.*

If (A activates B) is in $\alpha_{\mathcal{RI}}(x)$ but (A inhibits B) is not in $\alpha_{\mathcal{RI}}(x)$ then (A activates B) is in $\alpha_{\mathcal{JI}} \circ \beta(x)$.

If (A inhibits B) is in $\alpha_{\mathcal{RI}}(x)$ but (A activates B) is not in $\alpha_{\mathcal{RI}}(x)$ then (A inhibits B) is in $\alpha_{\mathcal{JI}} \circ \beta(x)$.

Proof. Since $\partial \dot{B}/\partial A = \sum_{i=1}^{n} (r_i(B) - l_i(B)) * \partial e_i/\partial A$ and all e_i are monotonic we get that $\partial \dot{B}/\partial A = \sum_{\{i \leq n | l_i(A) > 0\}} (r_i(B) - l_i(B)) * \partial e_i/\partial A$.

Now if (A activates B) is in $\alpha_{\mathcal{RI}}(x)$ but (A inhibits B) is not in $\alpha_{\mathcal{RI}}(x)$ then all rule such that $l_i(A) > 0$ verify $r_i(B) - l_i(B) \geq 0$ and there is at least one rule for which the inequality is strict. We thus get that $\partial \dot{B}/\partial A$ is a sum of positive numbers, amongst which one is such that $r_i(B) - l_i(B) > 0$ and $l_i(A) > 0$ which, since x is strongly monotonic, implies that there exists a point in the space for which $\partial e_i/\partial A > 0$ thus $\partial \dot{B}/\partial A > 0$ at that point and (A activates B) is in $\alpha_{\mathcal{JI}} \circ \beta(x)$.

For inhibition the same reasoning applies with the opposite sign for the $r_i(B) - l_i(B)$ and thus for the finale partial derivative. □

This lemma establishes the following equivalence result:

Theorem 25 *Let x be a reaction model with strongly monotonic kinetics and where no molecule is at the same time an activator and an inhibitor of the same target molecule, then $\alpha_{\mathcal{RI}}(x) = \alpha_{\mathcal{JI}} \circ \beta(x)$.*

This theorem shows that for standard kinetic expressions, the syntactical influences coincide with the differential influences based on the signs of the coefficients in the Jacobian matrix, when no molecule is at the same time an activator and an inhibitor of the same molecule. The theorem thus provides a linear time algorithm for computing the differential influences in these cases, simply by computing the syntactical influences. It shows also that the graph of differential influences is independent of the kinetic expressions:

Corollary 26 *The graph of differential influences of a reaction model of n rules with strongly monotonic kinetics is computable in time $O(n)$ if no molecule is at the same time an activator and an inhibitor.*

Corollary 27 *The graph of differential influences of a reaction model is independent of the kinetic expressions as long as they are strongly monotonic, if no molecule is at the same time an activator and an inhibitor.*

5.4 Evaluation Results

5.4.1 MAPK model.

Let us first consider the MAPK signalling model of [32]. Fig. 6 depicts the reaction graph as a bipartite graph with round boxes for molecules and rectangular boxes for rules. Fig. 7 depicts the inferred influence graph, where activation (resp. inhibition) is materialized by plain (resp. dashed) arrows. The graph layouts of the figures have been computed in BIOCHAM by the Graphviz suite [2].

Since this model verifies the hypotheses of corollary 25 we know that abstracting from the kinetics would give the same result.

Interestingly, this influence graph of the MAPK cascade exhibits inhibition feedback loops although in this model, the reaction graph is a pure cascade containing no feedback reaction. The interpretation of these inhibition feedback loops by sequestration in complexes at the different levels of the cascade is analyzed in [44]. The possibility to obtain (damped) oscillations in such "cascades" has been observed in [35] showing the relevance of our automatic analysis in this example.

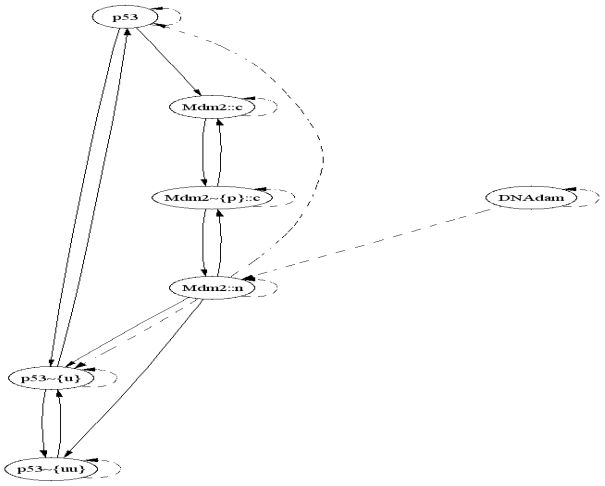

Fig. 1. Inferred influence graph of the p53-Mdm2 model

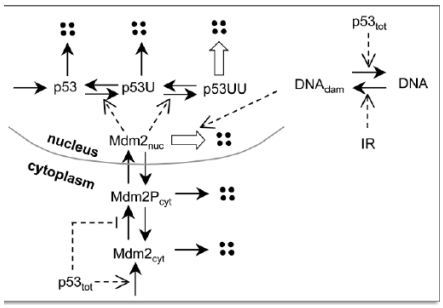

Fig. 2. Original reaction graph considered in [10] for the p53-Mdm2 model.

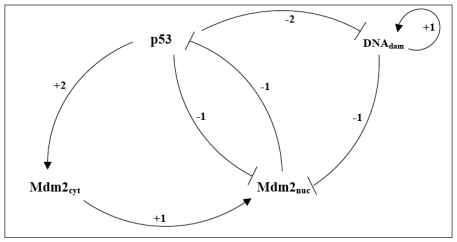

Fig. 3. Core influence graph[30].

5.4.2 p53-Mdm2 model.

In the p53-Mdm2 model of [10], the protein $Mdm2$ is localized explicitly in two possible locations: the nucleus and in the cytoplasm, and transport rules are considered. Fig. 2 depicts the reaction graph of the model.

Fig. 1 depicts the inferred influence graph. Note that $Mdm2$ in the nucleus has both an activation and an inhibitory effect on $p53 \sim \{u\}$. This corresponds to different influences in different regions of the space and one can check that

[2] `http://www.graphviz.org/`

the two influences also appear in $\alpha_{\mathcal{JI}}$.

Fig. 3 depicts the core influence graph considered for the logical analysis of this model [30]. In the core influence graph, some influence are neglected, as expected, however some inhibitions, such the inhibitory effect of $p53$ on $Mdm2$ in the nucleus, are considered while they do not appear in the inferred influence graph. The reason for these omissions is the way the reaction model is written. Some inhibitory effects are indeed expressed in the kinetic expression by subtraction of, or division by, the molecular concentration of some compounds that do not appear in the rule itself. Those non-monotonic inhibitions are thus missed by the type inference algorithm. An example of such a rule is the following one for the inhibition of $Mdm2$ by $p53$:

```
macro(p53tot,[p53]+[p53~{u}]+[p53~{uu}]).

(kph*[Mdm2::c]/(Jph+p53tot),MA(kdeph))
   for Mdm2::c <=> Mdm2~{p}::c.
```

Obviously, one cannot expect to infer such inhibitory effects from the reaction rules. Such a situation suggests to extend the syntax of reaction rules in order to indicate the inhibitors of the reaction, in a somewhat symmetric fashion to what is done for catalysts.

5.4.3 Kohn's Map.

On the quite big model of Kohn's map, the type inference of activation and inhibition influences from reaction rules takes less than one second CPU time (on a PC 1,7GHz) for the complete model, showing again the efficiency of the type inference algorithm.

As kinetic data is typically missing for such a large model, the influence analysis from the syntactical domain is the only one available.

6 A Type System for Location Topologies

To date, models of biochemical systems generally abstract from space considerations. Models taking into account cell compartments and transport phenomena are thus much less common. Nevertheless, with the advent of systems biology computational tools, more and more models are refined with space considerations and transport delays, e.g. [10]. In SBML [27] level 1 version 1, locations have been introduced as purely symbolic compartments without precise topology. We show in this section how the topology can be inferred

from the reaction rules, and checked in different models.

6.1 Abstract Domain of Location Topologies

We will here focus on the notion of *neighbor* that is supposed to represent the fact that two compounds live in two compartments that are next to each other. In SBML level 2, an *outside* relation can optionally be given for two compartments, stating that one is the outside of the other one. We should have, from our definition, that if A is the outside of B, then any compound living in A and any compound living in B are neighbors.

Definition 28 *The domain of neighborhood relations $\mathcal{D}_\mathcal{N} = \mathcal{P}(\mathcal{M} \times \mathcal{M})$ is about pairs of molecules. $\alpha_\mathcal{N} : \mathcal{D}_\mathcal{R} \to \mathcal{D}_\mathcal{N}$ is defined by the union of its definition on single rules:*

$\alpha_\mathcal{N}(\texttt{E for A}_1 + \cdots + \texttt{A}_n \texttt{ => B}_1 + \cdots + \texttt{B}_m)$ = All A_i and all B_j are pairwise neighbors, and for all C_k such that $\texttt{[C}_k\texttt{]}$ appears in $\texttt{E}$, C_k is a neighbor of all A_i and all B_j.

Proposition 29 *Let $\gamma_\mathcal{N}(n) = \cup {\alpha_\mathcal{N}}^{-1}(\downarrow n)$. $\mathcal{D}_\mathcal{R} \underset{\gamma_\mathcal{N}}{\overset{\alpha_\mathcal{N}}{\longleftrightarrow}}$ is a Galois connection.*

The concretization of a positive neighborhood between two locations is the set of all possible rules linking those compartments, i.e. transport rules or rules influencing one compartment from another one. It describes in some sense the interface between the two locations.

6.2 Evaluation Results

6.2.1 Models from `biomodels.net`.

We have taken models from the literature through the `http://www.biomodels.net` database. Of the 112 curated models in the current version (dated June 2007) only 35 have more than one compartment, and only 7 of those use the *outside* attribute of SBML to provide more topological insight.

The neighborhood relation is inferred in these models imported in BIOCHAM, and then checked consistent with the provided *outside* relation.

For instance for calcium oscillations, we tried both the Marhl et al. model of [33] and the Borghans et al. model of [2].

In the first case (model `BIOMD0000000039.xml`), three locations are defined: the cytosol, the endoplasmic reticulum and a mitochondria, from the reactions the inferred topology is that the cytosol is neighbor of the two other locations. This correspond exactly to the information obtained from the *outside* annotations (the cytosol being marked as the outside of the two other locations).

In the second case (models `BIOMD0000000043.xml` to `BIOMD0000000045.xml`) we focused on the last model (*two-pool*) since it is the only one with 4 different locations: the extracellular space, the cytosol and two internal vesiculae. The location inference produces a topology where the cytosol is neighbor of all other locations. Once again this is correct w.r.t. the outside information provided in the SBML file: both vesiculae have the cytosol as outside location and the cytosol itself has the extracellular space as outside location.

These considerations show that there is some mismatch between the SBML reaction models and the choice of expressing outside vs neighborhood properties of locations. In the perspective of type checking and type inference, neighborhood relations should be preferred as they can be checked, or inferred from the reaction model, whereas the outside relation contain more information that, while helpful for the modeler as meta-data, cannot be handled automatically without abstracting it first in neighbors properties. Note however that the SBML v. 3 effort rather goes in the opposite direction w.r.t. spatial information (see `http://sbml.org/wiki/Spatial_Features`) since it will allow a complete geometrical description of the compartments, which is of course very informative but is not amenable to automatic checking or inference.

Note also that in calculi where the topology of the network evolves, like the Brane calculus [6] and its derivatives, the outside and inside relationships change much more radically than the neighborhood relationship. For instance an exocytosis followed by an endocytosis might reverse the outside relationship whereas it would not change the neighborhood relation. Moreover, as shown in the second example below the neighborhood relation can easily be applied to cell (or compartment) populations to represent the topology, while defining only one "outside" for each cell makes the topology disappear.

6.2.2 P53/Mdm2.

The first example comes from [10]: a model of the p53/Mdm2 interaction with two locations (see Fig. 2) where the transport between cytoplasm and nucleus is necessary to explain some time delays observed in the mutual repression of these proteins.

```
biocham: load_biocham('EXAMPLES/locations/p53Mdm2.bc').
...
```

```
(MA(ko),MA(ki)) for Mdm2::n <=> Mdm2~{p}::c.
...
biocham: show_neighborhood.
c and n are neighbors
```

We restricted the output to the neighborhood between compartments rather than compounds for clarity.

In this precise case, the model as published does not systematically use the volume ratio in the kinetics. The transcription and type-checking of the model showed that if one wanted to keep the background degradation rate of $Mdm2$ (without DNA damage) independent of the location, one obtains different kinetics than those of the published model. In this case a formal transcription in BIOCHAM (or SBML) provided a supplementary model-validation step.

6.2.3 Delta and Notch Model.

The Delta and Notch proteins are crucial to the cell fate in different organisms. A population of neighboring cells is represented through locations, chosen here to be on a square grid. The model of Gosh and Tomlin [21] for the activation and inhibition of Delta and Notch proteins reproduce the salt-and-pepper coloring of the cells corresponding to high Delta-low Notch and low Delta-high Notch differentiation. This is typical of the Delta-Notch lateral inhibition based differentiation. The signaling pathways are simplified to the extreme to take into account only the direct effect of Delta and Notch expression in the cell and on the neighboring cells, with rules like :

```
biocham: load_biocham('EXAMPLES/locations/notch4n36c.bc').

(if [D::c21]+[D::c23]+[D::c12]+[D::c32] < 0.2
   then 0
   else ka,MA(kd)) for
      _ <=> N::c22.

(if [N::c22] > 0.5
   then 0
   else ka,MA(kd)) for
      _ <=> D::c22.

...
```

Note that in this example, as most of the information is in the kinetics of the rules, the analysis of influences should be done with the Jacobian of the differential semantics, instead of the syntactic domain of reaction rules, as described in section 5. However, for the analysis of location topology, the abstraction defined in this section provides the expected result, as depicted in

figure 4.

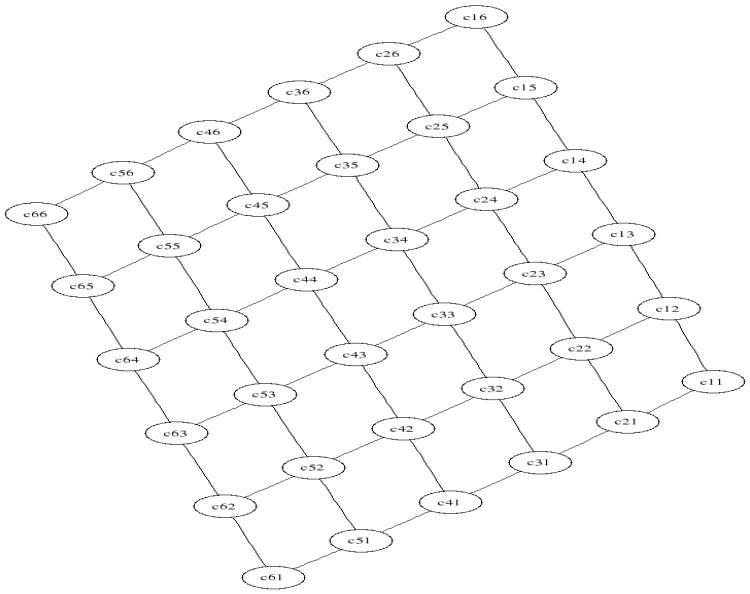

Fig. 4. Delta-Notch square cell grid inferred by $\alpha_{\mathcal{N}}$ in a 6x6 model.

This example also illustrates a subtlety in the definition of the abstraction function $\alpha_{\mathcal{N}}$. Indeed, it could be tempting to define the abstraction in the following simpler manner :

Definition 30 *$\alpha'_{\mathcal{N}} : \mathcal{D}_{\mathcal{R}} \rightarrow \mathcal{D}_{\mathcal{N}}$ is defined by the union of its definition on single rules:*

$\alpha'_{\mathcal{N}}$(`E for` $\mathtt{A}_1 + \cdots + \mathtt{A}_n$ `=>` $\mathtt{B}_1 + \cdots + \mathtt{B}_m$*) = All A_i, all B_j, and all C_k such that* `[`$\mathtt{C}_k$`]` *appears in* `E`*, are pairwise neighbors.*

Figure 5 depicts the topology inferred for Delta-Notch model with this second definition. It shows too coarse on such examples since co-modifiers are put in the kinetic expression of a single rule for simplification purposes. This illustrates the fact that lots of published reaction models rely extensively on the ODEs derived from the rules, the rules themselves being not always carefully written, but rather as compact as possible.

7 Conclusion

We have shown that the framework of abstract interpretation applies, on the one hand, to the organization of major semantics of biochemical reaction rules into a hierarchy of semantics related by abstraction functions, and on the other hand, to the formalization of some further abstractions commonly used in systems biology as type systems.

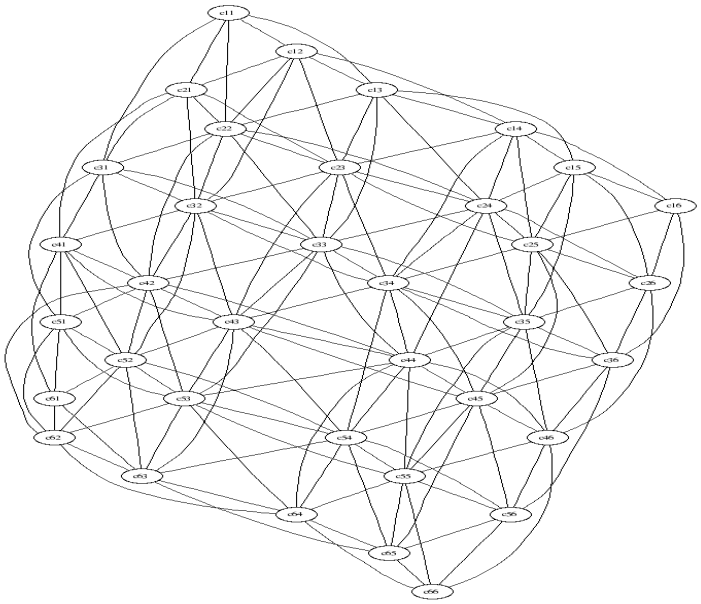

Fig. 5. Delta-Notch square cell grid inferred by $\alpha'_{\mathcal{N}}$ in a 6x6 model.

In the three type systems studied in this paper for, respectively, protein functions, activation and inhibitory influences, and location topologies, the analyses are based on static information gained directly from the syntax of reaction rules, without considering their formal semantics, nor their precise dynamics. It is worth noting that this situation also occurs in program analysis where the syntax of programs may capture a sufficient part of the semantics for many analyses. Here, it is remarkable that such simple analyses already provide useful information on biological models, independently from their dynamics for which different definitions are considered (discrete, continuous, stochastic, etc.).

The formal definition of the influence graph as an abstraction of the reaction model eliminates some confusion that exists in the use of Thomas's conditions [42,41] for the analysis of reaction models [34]. Such a formalization shows also that the influence graphs usually considered in the literature are further abstractions obtained by forgetting some influences, based on non-linearity considerations [43]. Some inhibitions may also be missing in the inferred influences when they are hidden in the kinetic expressions of the reactions and do not appear explicitly in the reactants. This suggests either to refine the abstraction function to take into account the kinetic expression when possible, or to extend the syntax of reactions in order to make explicit such inhibitory effects, in a symmetric fashion to catalysts for activations. In SBML there is actually an unique symmetrical notion of *Modifiers* which is not sufficient to infer the influence graph since it does not make any difference between activators and inhibitors.

Furthermore, we have shown that under general monotonicity conditions sat-

isfied by standard kinetics, such as the mass action law, Michaelis-Menten or Hill kinetics, the influences inferred from the syntax of reactants and products in the rules, include the influences inferred from the signs of the coefficient of the Jacobian matrix, and the equality holds when no molecule is both an activator and an inhibitor of a same molecule. This shows, perhaps surprisingly, that the Jacobian influences can be easily computed in linear time from the rule syntax, and that they are independent of the precise kinetic expressions under general conditions.

Similarly, the inference of protein functions and of location neighborhood have shown that the static analysis of reaction models by type inference provides both accurate and useful information. They also provide some guidelines for the extensions of biochemical reaction languages, like for instance in BIOCHAM, differentiating phosphorylation from other forms of modifications like acetylation, methylation, ubiquitination, etc. and in SBML, considering neighborhood rather than outside properties, and introducing a syntax for compound modifications.

Although the analyses done from the differential semantics of reaction rules have been compared to the analyses done from the syntax of reaction rules, the differential semantics itself is the only one that has not been related by Galois connections to the other semantics for several reasons explained in the corresponding section of this paper. These difficulties obviously provide an interesting subject for future work, from both the systems biology and the abstract interpretation theory standpoints.

Acknowledgement.

We gracefully acknowledge Monika Heiner for interesting discussions on the modeling of biochemical networks with Petri nets. This work benefited from partial support of the Network of Excellence REWERSE of the European Union and of the INRIA ARC MOCA.

References

[1] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[2] José Borghans, Geneviève Dupont, and Albert Goldbeter. Complex intracellular

calcium oscillations: a theoretical exploration of possible mechanisms. *Biophysical Chemistry*, 66:25–41, 1997.

[3] Laurence Calzone, Nathalie Chabrier-Rivier, François Fages, and Sylvain Soliman. Machine learning biochemical networks from temporal logic properties. In Gordon Plotkin, editor, *Transactions on Computational Systems Biology VI*, volume 4220 of *Lecture Notes in BioInformatics*, pages 68–94. Springer-Verlag, November 2006. CMSB'05 Special Issue.

[4] Laurence Calzone, François Fages, and Sylvain Soliman. BIOCHAM: An environment for modeling biological systems and formalizing experimental knowledge. *BioInformatics*, 22(14):1805–1807, 2006.

[5] Luca Cardelli. Typeful programming. In E. J. Neuhold and M. Paul, editors, *Formal Description of Programming Concepts*, pages 431–507. Springer-Verlag, Berlin, 1991.

[6] Luca Cardelli. Brane calculi - interactions of biological membranes. In Vincent Danos and Vincent Schächter, editors, *CMSB'04: Proceedings of the second international workshop on Computational Methods in Systems Biology*, volume 3082 of *Lecture Notes in BioInformatics*, pages 257–280. Springer-Verlag, 2004.

[7] Nathalie Chabrier and François Fages. Symbolic model cheking of biochemical networks. In Corrado Priami, editor, *CMSB'03: Proceedings of the first workshop on Computational Methods in Systems Biology*, volume 2602 of *Lecture Notes in Computer Science*, pages 149–162, Rovereto, Italy, March 2003. Springer-Verlag.

[8] Nathalie Chabrier-Rivier, Marc Chiaverini, Vincent Danos, François Fages, and Vincent Schächter. Modeling and querying biochemical interaction networks. *Theoretical Computer Science*, 325(1):25–44, September 2004.

[9] Claudine Chaouiya. Petri net modelling of biological networks. *Briefings in Bioinformatics*, 2007.

[10] Andrea Ciliberto, Béla Novák, and John J. Tyson. Steady states and oscillations in the p53/mdm2 network. *Cell Cycle*, 4(3):488–493, March 2005.

[11] Emmanuel Coquery and François Fages. Subtyping constraints in quasi-lattices. In P. Pandya and J. Radhakrishnan, editors, *Proceedings of the 23rd conference on foundations of software technology and theoretical computer science, FSTTCS'2003*, volume 2914 of *Lecture Notes in Computer Science*, pages 136–148, Mumbai, India, December 2003. Springer-Verlag.

[12] Patrick Cousot. Abstract interpretation and application to logic programs. *Journal of Logic Programming*, 13(2-3):103–179, 1992.

[13] Patrick Cousot. Types as abstract interpretation (invited paper). In *POPL'97: Proceedings of the 24th ACM Symposium on Principles of Programming Languages*, pages 316–331, New York, 1997. ACM Press. Paris.

[14] Patrick Cousot. Constructive design of a hierarchy of semantics of a transition system by abstract interpretation. *Theoretical Computer Science*, 277(1):47–103, 2002.

[15] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages*, pages 238–252, New York, 1977. ACM Press. Los Angeles.

[16] Steven Eker, Merrill Knapp, Keith Laderoute, Patrick Lincoln, José Meseguer, and M. Kemal Sönmez. Pathway logic: Symbolic analysis of biological signaling. In *Proceedings of the seventh Pacific Symposium on Biocomputing*, pages 400–412, January 2002.

[17] François Fages. From syntax to semantics in systems biology - towards automated reasoning tools. *Transactions on Computational Systems Biology IV*, 3939:68–70, December 2006.

[18] François Fages and Emmanuel Coquery. Typing constraint logic programs. *Journal of Theory and Practice of Logic Programming*, 1(6):751–777, November 2001.

[19] François Fages and Sylvain Soliman. Type inference in systems biology. In Corrado Priami, editor, *CMSB'06: Proceedings of the fourth international conference on Computational Methods in Systems Biology*, volume 4210 of *Lecture Notes in Computer Science*. Springer-Verlag, 2006.

[20] François Fages, Sylvain Soliman, and Nathalie Chabrier-Rivier. Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. *Journal of Biological Physics and Chemistry*, 4(2):64–73, October 2004.

[21] Ronojoy Ghosh and Claire Tomlin. Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In Springer-Verlag, editor, *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01*, volume 2034 of *Lecture Notes in Computer Science*, pages 232–246, Rome, Italy, 2001.

[22] Michael A. Gibson and Jehoshua Bruck. A probabilistic model of a prokaryotic gene and its regulation. In H. Bolouri and J.M. Bower, editors, *Computational Methods in Molecular Biology: From Genotype to Phenotype*, chapter 2. MIT press, 2000.

[23] David Gilbert, Monika Heiner, and Sebastian Lehrack. A unifying framework for modelling and analysing biochemical pathways using petri nets. In *CMSB'07: Proceedings of the fifth international conference on Computational Methods in Systems Biology*, volume 4695 of *Lecture Notes in Computer Science*. Springer-Verlag, 2007.

[24] Daniel T. Gillespie. General method for numerically simulating stochastic time evolution of coupled chemical-reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[25] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[26] William S. Hlavacek, James R. Faeder, Michael L. Blinov, Richard G. Posner, Michael Hucka, and Walter Fontana. Rules for modeling signal-transduction systems. *Science STKE*, 344:re6, 2006.

[27] Michael Hucka et al. The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[28] Trey Ideker, Timothy Galitski, and Leroy Hood. A new approach to decoding life: Systems biology. *Annual Review of Genomics and Human Genetics*, 2:343–372, 2001.

[29] Minoru Kanehisa and Susumu Goto. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30, 2000.

[30] Marcelle Kaufman. Private communication. 2006.

[31] Kurt W. Kohn. Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Molecular Biology of the Cell*, 10(8):2703–2734, August 1999.

[32] Andre Levchenko, Jehoshua Bruck, and Paul W. Sternberg. Scaffold proteins may biphasically affect the levels of mitogen-activated protein kinase signaling and reduce its threshold properties. *PNAS*, 97(11):5818–5823, May 2000.

[33] Marko Marhl, Thomas Haberichter, Milan Brumen, and Reinhart Heinrich. Complex calcium oscillations and the role of mitochondria and cytosolic proteins. *BioSystems*, 57:75–86, 2000.

[34] Nick I. Markevich, Jan B. Hoek, and Boris N. Kholodenko. Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. *Journal of Cell Biology*, 164(3):353–359, February 2005.

[35] Liang Qiao, Robert B. Nachbar, Ioannis G. Kevrekidis, and Stanislav Y. Shvartsman. Bistability and oscillations in the huang-ferrell model of mapk signaling. *PLoS Computational Biology*, 3(9):1819–1826, September 2007.

[36] V. N. Reddy, M. L. Mavrovouniotis, and M. N. Liebman. Petri net representations in metabolic pathways. In L. Hunter, D. B. Searls, and J. W. Shavlik, editors, *Proceedings of the 1st International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 328–336. AAAI Press, 1993.

[37] Andrea Sackmann, Monika Heiner, and Ina Koch. Application of petri net based analysis techniques to signal transduction pathways. *BMC Bioinformatics*, 7(482), November 2006.

[38] S. Schuster, T. Pfeiffer, F. Moldenhauer, and al. Exploring the pathway structure of metabolism: decomposition into subnetworks and application to mycoplasma pneumoniae. *Bioinformatics*, 18(3):51–61, 2002.

[39] Lee A. Segel. *Modeling dynamic phenomena in molecular and cellular biology.* Cambridge University Press, 1984.

[40] Sylvain Soliman and François Fages. CMBSlib: a library for comparing formalisms and models of biological systems. In Vincent Danos and Vincent Schächter, editors, *CMSB'04: Proceedings of the second international workshop on Computational Methods in Systems Biology*, volume 3082 of *Lecture Notes in BioInformatics*, pages 231–235. Springer-Verlag, 2004.

[41] Christophe Soulé. Graphic requirements for multistationarity. *ComplexUs*, 1:123–133, 2003.

[42] René Thomas, Anne-Marie Gathoye, and Lucie Lambert. A complex control circuit : regulation of immunity in temperate bacteriophages. *European Journal of Biochemistry*, 71(1):211–227, December 1976.

[43] René Thomas and Marcelle Kaufman. Multistationarity, the basis of cell differentiation and memory. *Chaos*, 11(1):170–195, March 2001.

[44] Alejandra C. Ventura, Jacques-Alexandre Sepulchre, and Soffia D. Merajver. A hidden feedback in signaling cascades is revealed. *PLoS Computational Biology*, to appear, 2008.

[45] I. Zevedei-Oancea and S. Schuster. Topological analysis of metabolic networks based on petri net theory. *In Silico Biology*, 3(29), 2003.

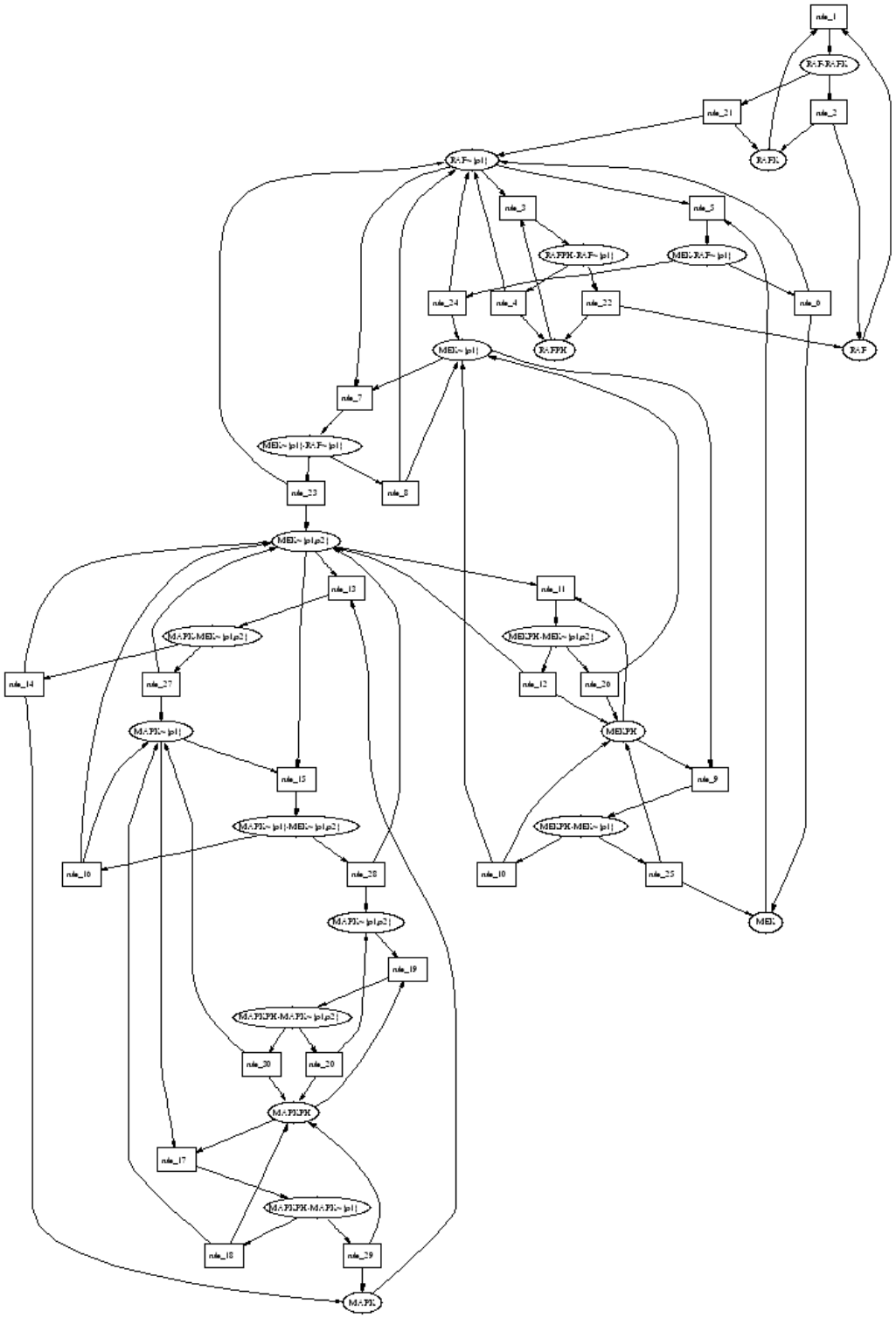

Fig. 6. Reaction graph of the MAPK cascade model

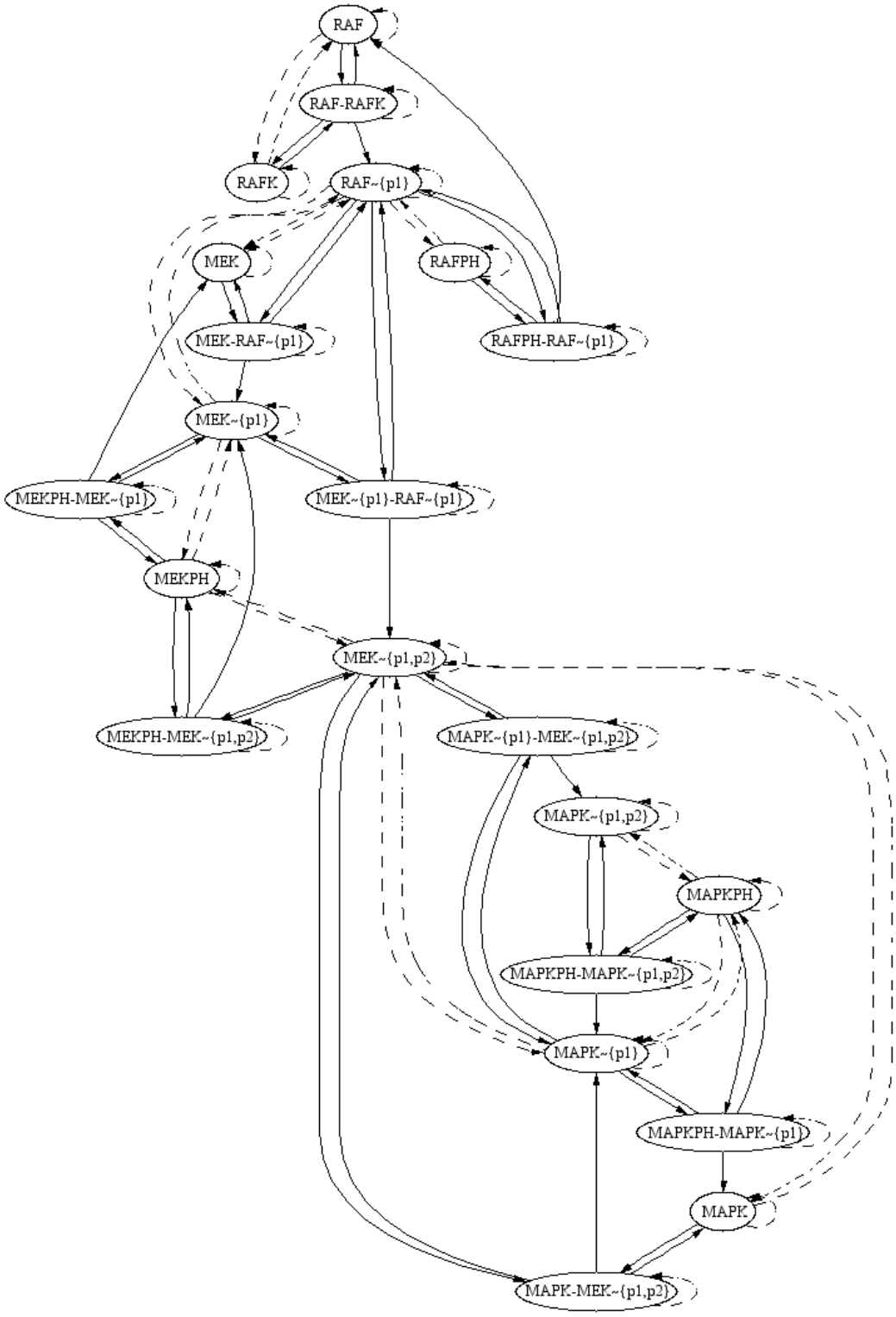

Fig. 7. Inferred influence graph of the MAPK cascade model showing negative feedback.